AMENDMENTS TO THE CLAIMS

1. (Currently Amended) A method comprising:

if a dependency exists between a greater width producer instruction and a lesser width consumer instruction, a processor substituting for execution a greater width source register ~~specifier~~ for a lesser width source register specified by the lesser width consumer instruction;

the processor executing the greater width producer instruction and placing a result of the execution in the greater width source register; and

the processor executing the lesser width consumer instruction using the greater width source register.

2. (Original) The method of claim 1 wherein the greater width source register substituted for the lesser width source register is the greater width register onto which the lesser width source register is aliased.

3. (Original) The method of claim 2, wherein substituting the greater width register includes setting an indication that the lesser width source register is to be replaced by the greater width register.

4. (Currently Amended) A method comprising:

if a dependency exists between a lesser width producer instruction and a greater width consumer instruction, a processor executing ~~substituting~~ plural instructions ~~for~~ instead of the greater width consumer instruction;

wherein the plural instructions substituted for the greater width consumer instruction include:

a first instruction to merge plural lesser width registers aliased onto a first greater width source register of the greater width consumer instruction, the plural lesser width registers to be merged into a first temporary register;

a second instruction to merge plural lesser width registers aliased onto a second greater width source register of the greater width consumer instruction, the plural lesser width registers to be merged into a second temporary register; and

a third instruction to execute the greater width consumer instruction using the first temporary register and the second temporary register as source registers.

5. (Canceled)

6. (Currently Amended) A method comprising:

determining, if a dependency exists between a greater width instruction and a lesser width instruction;

if a dependency exists between a greater width producer instruction and a lesser width consumer instruction, a processor substituting for execution a greater width source register ~~specifier~~ for a lesser width source register specified by the lesser width consumer instruction, executing the greater width producer instruction and placing an execution result into the greater width source register, and executing the lesser width consumer instruction using the greater width source register; and

if a dependency exists between a lesser width producer instruction and a greater width consumer instruction, the processor substituting plural instructions for the greater width consumer instruction and executing the plural instructions instead of the greater width consumer instruction.

7. (Currently Amended) The method of claim 6 further including stalling at least one instruction of a fetch group if a dependency exists between an instruction in the fetch group and both an active lesser width producer instruction and an active greater width producer instruction, until at least one of the active lesser width producer instruction or the active greater width producer instruction is retired, and then resuming execution.

8. (Original) The method of claim 6 wherein the greater width source register substituted for the lesser width source register is the greater width register onto which the lesser width source register is aliased.

9. (Original) The method of claim 8, wherein substituting the greater width register includes setting an indication that the lesser width source register is to be replaced by the greater width register.

10. (Original) The method of claim 6, wherein the plural instructions substituted for the greater width consumer instruction include:

a first instruction to merge plural lesser width registers aliased onto a first greater width source register of the greater width consumer instruction, the plural lesser width registers to be merged into a first temporary register;

a second instruction to merge plural lesser width registers aliased onto a second greater width source register of the greater width consumer instruction, the plural lesser width registers to be merged into a second temporary register; and

a third instruction to execute the greater width consumer instruction using the first temporary register and the second temporary register as source registers.

11. (Original) The method of claim 6, wherein determining if a dependency exists includes:

generating a first register mask identifying registers to be modified by lesser width instructions active in a pipeline; and

generating a second register mask identifying registers to be modified by greater width instructions active in the pipeline.

12. (Original) The method of claim 11, wherein determining if a dependency exists includes:

comparing a lesser width register specifier of an instruction against the second register mask; and

comparing a greater width register specifier of an instruction against the first register mask.

13. (Original) The method of claim 6, wherein determining if a dependency exists includes determining if a greater width instruction in a fetch group modifies a lesser width source register specified by a younger instruction in the same fetch group.

14. (Original) The method of claim 6, wherein determining if a dependency exists includes determining if a lesser width instruction in a fetch group modifies a greater width source register specified by a younger instruction in the same fetch group.

15. (Currently Amended) A method of handling a register conflict between a first instruction specifying a greater width destination register and a second instruction specifying a lesser width source register, the method comprising ~~substituting for execution~~ a processor executing the second instruction using the greater width destination register as a source instead of the lesser width source register ~~specifier of the second instruction with a greater width source register specifier~~.

16. (Original) The method of claim 15, wherein the register specified by the greater width source specifier is a greater width register onto which the register specified by the lesser width source register specifier is aliased.

17. (Original) The method of claim 15, wherein substituting the greater width register specifier includes setting an indication that the register specified by the lesser width source register specifier is to be replaced by the greater width register.

18. (Currently Amended) A method of handling a register conflict between a first instruction specifying a lesser width destination register and a second instruction specifying a greater width source register, the method comprising:

a processor substituting plural instructions for the second ~~instruction, wherein the plural instructions include:~~ instruction, the plural instructions comprising a first substitute instruction and a second substitute instruction;

the processor executing at least [[a]] the first substitute instruction to merge plural lesser width registers aliased onto the source register specified by the second instruction into a temporary register; and

the processor executing [[a]] the second substitute instruction to perform the operation specified by the second instruction using the temporary register as a source register; and

the processor executing a third substitute instruction to merge plural lesser width registers aliased onto a second greater width source register specified by the second instruction into a second temporary register, and wherein the second substitute instruction performs the operation using the first temporary register and the second temporary register.

19. (Canceled).

20. (Original) An instruction decode unit including:

logic to substitute a greater width source register specifier for a lesser width source register specifier if a dependency exists between a greater width producer instruction and a lesser width consumer instruction.

21. (Original) The instruction decode unit of claim 20 further including logic to prevent at least one instruction from being delivered for execution if a dependency exists between an instruction in the fetch group and both an active lesser width producer instruction and an active greater width producer instruction.

22. (Original) The instruction decode unit of claim 20, wherein the logic to substitute the greater width register specifier includes logic to set an indicator in the lesser width source register specifier indicating to that the lesser width source register is to be replaced by the greater width register.

23. (Currently Amended) An instruction decode unit including:

logic to substitute plural instructions for a greater width consumer instruction if a dependency exists between a lesser width producer instruction and a greater width consumer instruction;

wherein the logic to substitute plural instructions for the greater width consumer instruction includes:

logic to generate a first instruction to merge plural lesser width registers aliased onto a first greater width source register specified by the greater width consumer instruction, the plural lesser width registers being merged into a first temporary register;

logic to generate a second instruction to merge plural lesser width registers aliased onto a second greater width source register specified by the greater width consumer instruction, the plural lesser width registers being merged into a second temporary register; and

logic to designate the first temporary register and the second temporary register as source registers of the greater width consumer instruction.

24. (Canceled).

25. (Previously Presented) A processor comprising:

an instruction decode unit including:

logic to substitute a greater width source register specifier for a lesser width source register specifier if a dependency exists between a greater width producer instruction and a lesser width consumer instruction, and

logic to substitute plural instructions for a greater width consumer instruction if a dependency exists between a lesser width producer instruction and a greater width consumer instruction.

26. (Currently Amended) The processor of claim 25 further including logic to prevent at least one instruction from being delivered for execution if a dependency exists between an

instruction in the fetch group and both an active lesser width producer instruction and an active greater width producer instruction, until at least one of the active lesser width producer instruction or the active greater width producer instruction is retired, and then resuming execution.

27. (Original) The processor of claim 25, wherein the logic to substitute the greater width register specifier includes logic to set a bit in the lesser width source register specifier indicating to that the lesser width source register is to be replaced by the greater width register.

28. (Original) The processor of claim 25, wherein the logic to substitute plural instructions for the greater width consumer instruction includes:

logic to generate a first instruction to merge plural lesser width registers aliased onto a first greater width source register specified by the greater width consumer instruction, the plural lesser width registers being merged into a first temporary register;

logic to generate a second instruction to merge plural lesser width registers aliased onto a second greater width source register specified by the greater width consumer instruction, the plural lesser width registers being merged into a second temporary register; and

logic to designate the first temporary register and the second temporary register as source registers of the greater width consumer instruction.

29. (Original) The processor of claim 25, further including logic to determine if a greater width producer instruction that is part of a fetch group modifies a lesser width source register specified by a younger instruction in the same fetch group.

30. (Original) The processor of claim 25, further including logic to determine if a lesser width producer instruction that is part of a fetch group modifies a greater width source register specified by a younger instruction in the same fetch group.

31. (Original) A processor comprising

means for determining if a dependency exists between a greater width instruction and a lesser width instruction;

means for substituting a greater width source register specifier for a lesser width source register specifier if a dependency exists between a greater width producer instruction and a lesser width consumer instruction,; and

means for substituting plural instructions for the greater width consumer instruction if a dependency exists between a lesser width producer instruction and a greater width consumer instruction.

32. (Currently Amended) The processor of claim 31, further including means for stalling at least one instruction in a fetch group if a dependency exists between an instruction in the fetch group and both an active lesser width producer instruction and an active greater width producer instruction, until at least one of the active lesser width producer instruction or the active greater width producer instruction is retired, and then resuming execution.

33. (Original) The processor of claim 31, wherein the means for substituting plural instructions for the greater width consumer instruction include:

means for generating a first instruction to merge plural lesser width registers aliased onto a first greater width source register of the greater width consumer instruction, the plural lesser width registers being merged into a first temporary register;

means for generating a second instruction to merge plural lesser width registers aliased onto a second greater width source register of the greater width consumer instruction, the plural lesser width registers being merged into a second temporary register; and

means for generating a third instruction to execute the greater width consumer instruction using the first temporary register and the second temporary register as source registers.

34. (Original) The processor of claim 31, wherein the means for determining if a dependency exists includes means for determining if a greater width destination instruction in a fetch group modifies a lesser width source register specified by a younger instruction in the same fetch group.

35. (Original) The processor of claim 31, wherein the means for determining if a dependency exists includes means for determining if a lesser width instruction in a fetch group modifies a greater width source register specified by a younger instruction in the same fetch group.

36. (Previously Presented) The processor of claim 25, wherein the instruction decode unit also includes: plural counters to track a number of instructions active in a pipeline;

plural mask registers to hold vectors indicating particular registers to be modified by active instructions; and

logic to compare destination and source registers specified by the instructions against at least one of the plural mask registers to determine if a dependency exists between an instruction being decoded and an active instruction.